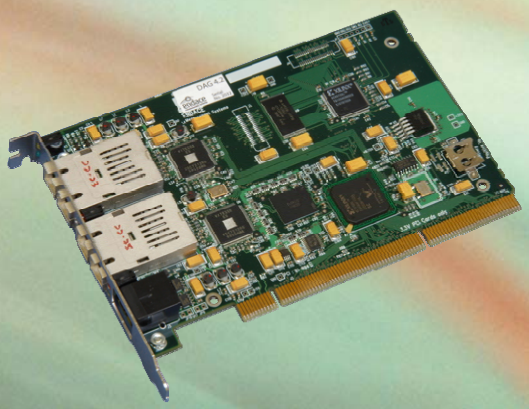
# DAG 4.2GE Card User Manual
2.5.5r1

EDM01.05-15r1



Endace Measurement Systems | New Zealand | www.endace.com

# Leading Network Intelligence



Published by:

Endace Measurement Systems® Ltd
Building 7
17 Lambie Drive
PO Box 76802
Manukau City 1702
New Zealand
Phone: +64 9 262 7260
Fax: +64 9 262 7261
support@endace.com
www.endace.com

**International Locations**

**New Zealand**
Endace Technology® Ltd
Level 9
85 Alexandra Street
PO Box 19246
Hamilton 2001
New Zealand
Phone: +64 7 839 0540
Fax: +64 7 839 0543
support@endace.com
www.endace.com

**Americas**
Endace USA® Ltd
Suite 220
11495 Sunset Hill Road
Reston
Virginia 20190
United States of America
Phone: ++1 703 382 0155
Fax: ++1 703 382 0155
support@endace.com
www.endace.com

**Europe, Middle East & Africa**
Endace Europe® Ltd
Sheraton House
Castle Park
Cambridge CB3 0AX
United Kingdom
Phone: ++44 1223 370 176
Fax: ++44 1223 370 040
support@endace.com
www.endace.com

## Typographical Conventions Used in this Document

- Command-line examples suitable for entering at command prompts are displayed in `mono-space courier font`.

  Results generated by example command-lines are also displayed in `mono-space courier font`.

- The software version references such as 2.3.x, 2.4.x, 2.5.x are specific to Endace Measurement Systems and relate to Company software products only.

## Protection Against Harmful Interference

When present on product this manual pertains to and indicated by product labelling, the statement "This device complies with part 15 of the FCC rules" specifies the equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the Federal Communications Commission [FCC] Rules.

These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment.

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications.

Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

## Extra Components and Materials

The product that this manual pertains to may include extra components and materials that are not essential to its basic operation, but are necessary to ensure compliance to the product standards required by the United States Federal Communications Commission, and the European EMC Directive. Modification or removal of these components and/or materials, is liable to cause non compliance to these standards, and in doing so invalidate the user’s right to operate this equipment in a Class A industrial environment.

## USE THIS SPACE FOR NOTES

# Table of Contents

## USE THIS SPACE FOR NOTES

# 1.0 PREFACE

**Introduction** The installation of the Endace DAG 4.2GE card on a PC begins with installing the operating system and the Endace software. This is followed by fitting the card and connecting the ports.

**Viewing this document** This document, DAG 4.2GE Card User Manual is available on the installation CD.

**In this chapter** This chapter covers the following sections of information.

- User Manual Purpose
- DAG 4.2GE Card Product Description
- DAG 4.2GE Card Architecture
- DAG 4.2GE Card Extended Functions
- DAG 4.2GE Card System Requirements

## 1.1 User Manual Purpose

**Description** The purpose of this DAG Card User Manual is to describe:

- Installing DAG 4.2GE card
- Setting DAG 4.2GE Card Optical Power
- Confidence Testing DAG 4.2GE Card
- Running Data Capture Software
- Synchronizing Clock Time
- Data Formats Overview

**Pre-requisite** This document presumes the DAG card is being installed in a PC already configured with an operating system.

A copy of the Debian Linux 3.1 (Sarge) is available as a bootable ISO image on one of the CD's shipped with the DAG card.

To install on the Linux/FreeBSD operating system, follow the instructions in the document EDM04.05-01r1 Linux FreeBSD Installation Manual, packaged in the CD shipped with the DAG card.

To install on a Windows operating system, follow the instructions in the document EDM04.05-02r1 Windows Installation Manual, packaged in the CD shipped with the DAG card.

## 1.2 DAG 4.2GE Card Product Description

**Description** The DAG cards are PCI-bus cards designed for cell and packet capture and generation, specialised on IP networks. This document describes the installation and confidence testing of the DAG 4.2GE dual interface 1000baseSX Gigabit Ethernet cards.

**Figure** Figure 1-1 shows the DAG 4.2GE card.

Figure 1-1. DAG 4.2GE Card.

## 1.3 DAG 4.2GE Card Architecture

**Description** Serial Ethernet optical data is received by the two 1000baseSX optical interfaces, and passed through demultiplexors.

The network data is then fed immediately into the Xilinx FPGA. This FPGA contains the DUCK timestamp engine, packet record processor, and PCI interface logic. The close association of these components means that packets or cells can be time-stamped very accurately.

Time stamped packet records are then stored in an external FIFO before transmission to the host.

*Continued on next page*

## 1.3 DAG 4.2GE Card Architecture, continued

**Figure** Figure 1-2 shows the DAG 4.2GE card major components and data follow.

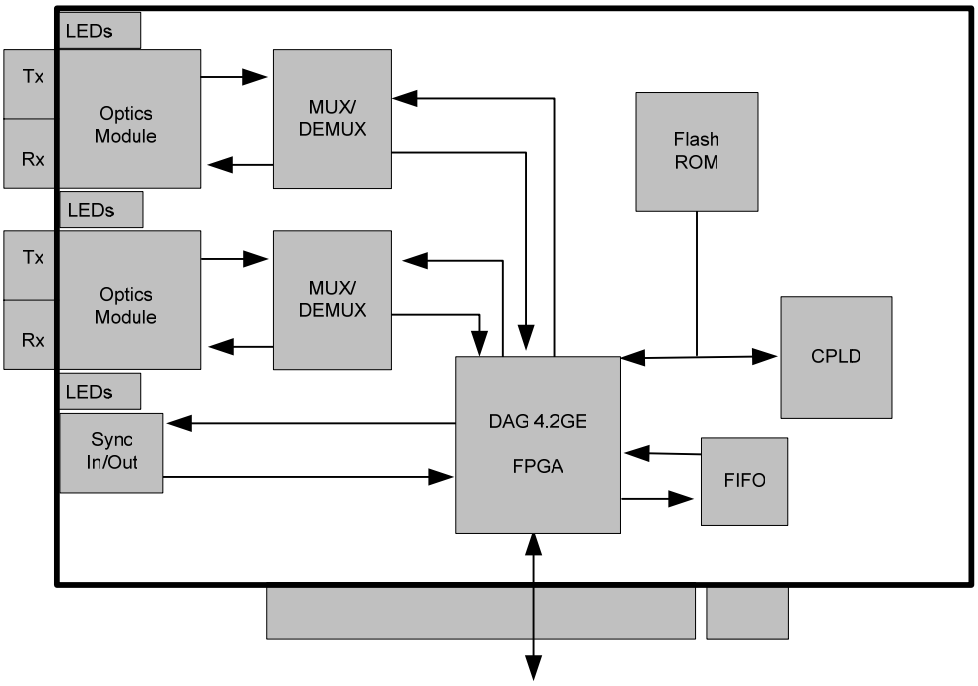



Figure 1-2. DAG 4.2GE Card Major Components and Data Flow.

## 1.4 DAG 4.2GE Card Extended Functions

**Description** The functionality of the DAG 4.2GE can be extended in many ways. A physical transmit path is provided on the DAG 4.2GE so packet generation is possible, but this requires special FPGA images.

To discuss the use of extended features contact support@endace.com

## 1.5 DAG 4.2GE Card System Requirements

**Description** The DAG 4.2GE and its associated data capture system has the following minimum system requirements:

- PC, at least Pentium III 800MHz or faster
- Intel i840, ServerWorks III LE/HE or newer chip set
- Minimum of 128 MB RAM
- At least one free 64-bit 3.3v signaling only PCI slot with 3.3V and 5V power
- Software distribution requires 30MB free space

*Continued on next page*

## 1.5 DAG 4.2GE Card System Requirements, continued

**Operating system**

For convenience, a Debian 3.1 [Sarge] Linux system is included on the Endace Software Install CD. Endace currently supports Windows XP, Windows Server 2000, Windows Server 2003, FreeBSD, RHEL 3.0, and Debian Linux operating systems.

**Different system**

For advice on using a system substantially different from that specified above, contact Endace support at support@endace.com

# 2.0 INSTALLING DAG 4.2GE CARD

**Introduction**

A DAG 4.2GE card can be installed in any free 3.3v signalling 64-bit Bus Mastering PCI slot.

By default, the driver supports up to four DAG cards in one system, but it is not recommended to have more than 2 cards on a single PCI bus due to bandwidth limitations, as the cards make very heavy use of PCI bus data transfer resources.

However, this is not usually a limitation as for most applications a maximum of two cards only can be used with reasonable application performance.

**In this chapter**

This chapter covers the following sections of information:

- Installation of Operating System and Endace Software
- Insert DAG 4.2GE Card into PC
- DAG 4.2GE Card Port Connectors

## 2.1 Installation of Operating System and Endace Software

**Description**

If the DAG device driver is not installed, before proceeding with the next chapter, install the software by following the instructions in EDM04-01 Endace Software Installation Manual.

Go to the next chapter of information when the DAG device driver is installed.

## 2.2 Insert DAG 4.2GE Card into PC

**Description**

Inserting the DAG 4.2GE card into a PC involves accessing the PCI-X bus slot, fitting the card, and secure the bus slot screw.

**Procedure**

Follow these steps to insert the DAG 4.2GE card into a PC.

**Step 1. Access bus Slot**

Power computer down.

Remove PCI bus slot cover.

**Step 2. Fit Card**

Insert DAG 4.2GE card into PCI-X bus slot.

*Continued on next page*

## 2.2 Insert DAG 4.2GE Card into PC, continued

**Procedure**, continued

**Step 3. Replace bus Slot Screw**

Secure card with screw.

**Step 4. Power Up Computer**

## 2.3 DAG 4.2GE Card Port Connectors

**Description**

There are two pairs of SC-type optical connectors. The bottom connector of each pair, nearest the PCI slot, is for the received signal, the top for the transmitted signal.

The transmit port is only connected if the loop back facility in the DAG card is used to daisy chain systems, or if the a data generation program is being used. The lower pair is called Port B, the upper pair is called Port A.

If the Tx port of the DAG card is not used, the SC-type transceiver optics should be covered to prevent ingress of dust.

An 8-pin RJ45 socket is used for time synchronization. This socket should never be connected to an Ethernet network or telephone line.

# 3.0 SETTING DAG 4.2GE CARD OPTICAL POWER

**Description**

The optical power range depends on the particular device fitted on the DAG 4.2GE card.

The DAG 4.2GE is shipped fitted with two 10000baseSX HFBF 53A5VFM 850nm multi-mode short range optics modules by default.

**Optical power measure**

Optical power is measured in dBm – decibels relative to 1 mW where 10 dB is equivalent to a factor of 10 in power.

The numbers are all negative, showing powers below 1 mW. The most sensitive devices can work down to about –30 dBm, or 1 uW.

**Configuration**

Table 3-1 shows the DAG 4.2GE card optics power module configuration.

| Part # | Fibre | Data Rate | Max Power [dBm] | Min Power [dBm] | Nominal Pwr [dBm] |
|---|---|---|---|---|---|
| HFBR53AVFM | MMF | 1000 | 0 | -17 | -14 |

**Definitions**

MMF: Multi-Mode Fibre. SMF: Single-Mode Fibre.

**In this chapter**

This chapter covers the following sections of information:

- Interpreting DAG 4.2GE Card LED Status
- DAG 4.2GE Card LED Display Functions

## 3.1 Optical Power Input

**Description**

The optical power input to the DAG 4.2GE card must be within the receiver's dynamic range of 0 to -22dBm.

When optical power is slightly out of range an increased bit error rate is experienced. If power is well out of range the system cannot lock onto the Ethernet signal. In extreme cases of being out range excess power will damage a receiver.

When power is above the upper limit the optical receiver saturates and fails to function. When power is below the lower limit the bit error rate increases until the device is unable to obtain lock and fails.

**Input power**

When the DAG 4.2GE card is set up, measure the optical power at the receiver and ensure that it is well within the specified power range.

Input power is adjusted by:

- Changing splitter ratio if power is too high or too low, or
- Inserting an optical attenuator if power is too high.

*Continued on next page*

## 3.2 Splitter Losses

**Description**

Splitters have the insertion losses marked on packaging or in accompanying documentation.

- A 50:50 splitter will have an insertion loss of between 3 dB and 4 dB on each output
- 90:10 splitter will have losses of about 10 dB in the high loss output, and <2 dB in the low loss output

The 1000baseSX transceiver uses 850nm optics. Splitters used must be designed for 850nm as the insertion loss will vary for different wavelengths.

**Single mode fibre loss**

A single mode fibre connected to a multi-mode input has minimal extra loss.

**Multi-mode fibre loss**

A multi-mode fibre connected to a single mode input creates large and unpredictable loss.

# 4.0 CONFIDENCE TESTING DAG 4.2GE CARD

**Introduction**

The confidence testing is a process to determine the DAG 4.2GE card is functioning correctly.

The process also involves a card capture session, and demonstrates configuration in the style of 'What You See You Can Change', WYSYCC.

Interface statistics are also inspected during this process.

**In this chapter**

This chapter covers the following sections of information.

- Interpreting DAG 4.2GE Card LED Status
- DAG 4.2GE Card LED Display Functions
- Configuration in WYSYCC Style
- DAG Card Capture Session
- Inspect Interface Statistics
- Reporting Problems

## 4.1 Interpreting DAG 4.2GE Card LED Status

**Description**

The DAG 4.2GE has 6 status LED's; one coloured blue, two green, one orange, and two red.

When a DAG 4.2GE card is powered up the blue LED 1 should always come on. The other LED's display for specific functions.

**Figure**

Figure 4-1 shows the DAG 4.2GE card LED status.

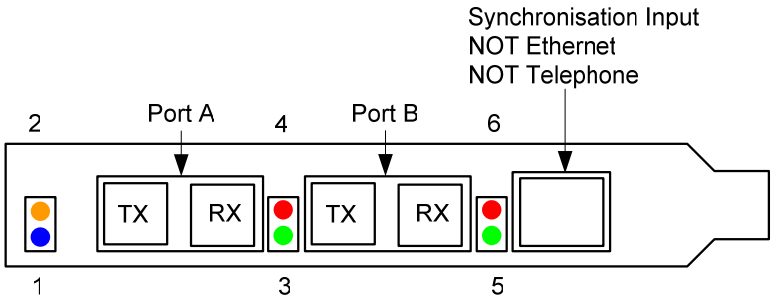



Figure 4-1. DAG 4.2GE LED Status.

*Continued on next page*

## 4.1 Interpreting DAG 4.2GE Card LED Status, continued

**LED definitions** The following table describes the LED definitions.

| LED | Description |
|---|---|
| LED 1 | Blue, FPGA successfully programmed. |
| LED 2 | Yellow, Data capture in progress. |
| LED 3 | Green, Port A Signal Detect – valid optical signal seen by the optical receiver. |
| LED 4 | Red, Port A link down. |
| LED 5 | Green, Port B Signal Detect; valid optical signal seen by optical receiver. |
| LED 6 | Red, Port B link down. |

**LED 4 and 6 indications** If Port A is in *nic* mode, this indicates the port failed to autonegotiate with the directly connected equipment. If Port B is in *nonic* mode, the link being monitored is down. The link endpoints have not autonegotiated correctly.

## 4.2 DAG 4.2GE Card LED Display Functions

**Description** When a DAG 4.2GE series card is powered up the blue LED 1 should always come on. The DAG 4.2GE card defaults to *nic* mode.

**LED definitions** The following table describes the DAG 4.2GE card LED definitions when a packet capture session is in progress.

| LED | Indicates |
|---|---|
| LED 1 | Blue, FPGA successfully programmed. |
| LED 2 | Data capture in progress. |
| LED 3 | Port A Signal Detect – valid optical signal seen by the optical receiver. |
| LED 4 | Link Down condition in *nic* mode or Monitored Link Down condition in *nonic* mode on Port A. |
| LED 5 | Port B Signal Detect – valid optical signal seen by the optical receiver. |
| LED 6 | Link Down condition in *nic* mode or Monitored Link Down condition in *nonic* mode on Port B. |

**Figure** Figure 4-2 shows the DAG 4.2GE card correct LED status without optical input.

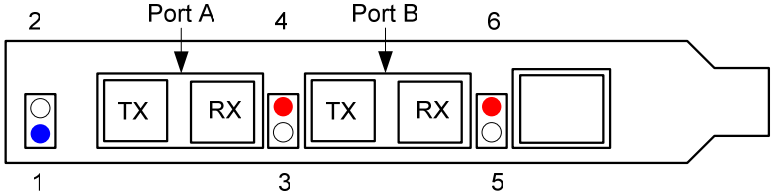



Figure 4-2. DAG 4.2GE Card Correct LED Status Without Optical input.

## 4.3 Card Configuration

**Description** A successful DAG 4.2GE card capture session is accomplished by checking receiver ports optical signal levels and checking the card has correctly detected the link. This is followed by configuring the DAG card for normal use.

**Procedure** Follow these steps for the DAG 4.2GE card configuration.

**Step 1. Check Receiver Ports Optical Signal Levels**

The card supports 850 nanometer multimode fibre attachments with optical signal strength between 0 dBm and -17 dBm.

If in doubt, check card receiver ports light levels are correct using an optical power meter.

The card receiver ports are the lower of each dual-SC-style connectors, the closest to the PCI slot. Cover unused ports with SC-style plugs to prevent dust and mechanical hazards from damaging optics.

**Step 2. Check FPGA Image Loaded.**

Before configuring the card, ensure the most recent FPGA image is loaded on the card.

```
dag@endace:~$ dagrom -rvp –d dag0 < xilinx/dag423eth-erf.bit
dag@endace:~$ dagfour –d/dev/dag0
linkA   noreset nonic nofcl noeql enableA
linkB   noreset nonic nofcl noeql enableB
packet  varlen slen=1536
packetA drop=0
packetB drop=0
pci     66MHz 64-bit rxonly
```

**Step 3. Configure DAG 4.3GE Card for Normal Use**

Configure the card according to the local settings, and check through the physical layer statistics that the card is locked to the data stream.

NOTE: For both configuration and statistics of the DAG 4.2GE framer the `dagfour` tool is supplied. Calling `dagfour` without arguments will list the current settings. `dagfour -h` will print a help listing on the usage of the tool.

## 4.4 Configuration in WYSYCC Style

**Description**

Configuration in WYSYCC is the 'What You See You Can Change' style. For example, if the facility loopback is turned off a user turns it on by typing:

```
dag@endace:~$ dagfour -d dag0 fcl
linkA   noreset nonic fcl noeql enableA
linkB   noreset nonic fcl noeql enableB
packet  varlen slen=1536
packetA drop=0
packetB drop=0
pci     66MHz 64-bit rxonly
```

**Ports**

Commands are applied to both ports by default. To affect only one port, use the `-a` or `-b` options. To disable a port for capturing, use the `disableA` and `disableB` commands.

**Auto-negotiate**

The DAG 4.2GE card can operate in one of two modes, nic and nonic.

The nic mode assumes that the card is connected directly to a Gigabit Ethernet switch or card with a full-duplex cable and the DAG will perform Gigabit Ethernet auto-negotiation.

The nonic mode is intended for use with optical fibre splitters. The receive socket of the DAG card port is connected to the output of an optical splitter that is inserted into a network link between two other devices, and the transmit socket of the DAG card is unconnected.

In this mode, Gigabit Ethernet auto-negotiation is not performed. One splitter on each DAG card receive port can then be used to monitor each direction of a full-duplex Gigabit Ethernet link.

*Continued on next page*

## 4.4 Configuration in WYSYCC Style, continued

**Configuration options**

The configurations supported are listed in the following table.

| | |
|---|---|
| `default` | set card to normal defaults. |
| `reset` | reset the physical layer. |
| `[no]nic` | [un]set nic mode. |
| `[no]fcl` | [un]set facility loop back in the phy. This is useful for card chaining. |
| `[no]eql` | [un]set equipment loop back in the phy. |
| `(en\|dis)ableA` | enable or disable port A for capture. |
| `(en\|dis)ableB` | enable or disable port B for capture. |
| `[no]varlen` | dis/enable variable length capture. Otherwise record length padded to slen. |
| `slen=X` | capture packets of X bytes long. |

## 4.5 Inspect Interface Statistics

**Description**

Once the card has been configured, the interface statistics are inspected to check the card is locked to the data stream.

```
dag@endace:~$ dagfour -d dag0 -si
```

The tool displays a number of status bits that have occurred since last reading. The following example shows the interval is set to one second, default value, via the `-i` option.

**Status bits**

| | |
|---|---|
| `LLU` | Local Link Up.<br>in *nic* mode, this indicates the card has negotiated a valid link state. |
| `PLU` | Peer Link Up.<br>This indicates that the peer's state is link up. |
| `LoS` | Loss of Signal.<br>No valid optical signal is being received. |
| `LoF` | Loss of Framing.<br>No valid Gigabit Ethernet signal has been detected. |
| `LCW` | Link Configuration Word.<br>Auto-negotiation word received from peer. |

*Continued on next page*

## 4.5 Inspect Interface Statistics, continued

**Status bits**, continued

**Example** The following is an example of the DAG 4.2GE in nic mode, with a Gigabit Ethernet NIC connected to port B via a full-duplex cable. Port A is unconnected.

```
dag@endace:~$ dagfour –d dag0 –si
Port A:   LLU   PLU   LoS   LoF   LCW    Port B:   LLU   PLU   LoS   LoF   LCW
            0     0     1     1   4020              1     1     0     0   01a0
            0     0     1     1   4020              1     1     0     0   01a0
            0     0     1     1   4020              1     1     0     0   01a0
            0     0     1     1   4020              1     1     0     0   01a0
```

**Extended statistics** Extended statistics are also available. Here is an example showing extended statistics from port B only for the above configuration.

```
 dag@endace:~$  dagfour –d dag1 –bei
Port B:   LLU   PLU   LoS   LoF   LCW    Bad-Symb   CRC-Fail  Remote-I  Bytes  Frames
            1     1     0     0   01a0          0          0         0      0       0
            1     1     0     0   01a0          0          0         0      0       0
            1     1     0     0   01a1          0          0         0      0       0
            1     1     0     0   01a1          0          0         0      0       0
```

The extended statistics include:

`Bad-Symb` Bad Symbol Counter.
Counts the number of invalid Gigabit Ethernet Symbols received.

`CRC-Fail` Local CRC Failure Counter.
Number of Ethernet frames received that failed CRC check.

`Remote-I` Remote Indications Counter
The peer is propagating an error condition, a jammed or corrupted packet for example.

`Bytes` Received Byte Counter.

`Frames` Received Frame Counter.

## 4.6 Reporting Problems

**Description**

If there are unresolved problems with a DAG card or supplied software, contact Endace Technical Support via the email address support@endace.com. Supplying sufficient information in an email enables effective response.

**Problem checklist**

The exact information available to users for trouble, cause and correction analysis may be limited by nature of the problem. The following items assist a quick problem resolution:

| Ref | Item |
|---|---|
| 1. | DAG card[s] model and serial number. |
| 2. | Host PC type and configuration. |
| 3. | Host PC operating system version. |
| 4. | DAG software version package in use. |
| 5. | Any compiler errors or warnings when building DAG driver or tools. |
| 6. | For Linux and FreeBSD, messages generated when DAG device driver is loaded. These can be collected from command `dmesg` or from log file `/var/log/syslog`. |
| 7. | Output of `daginf`. |
| 8. | Firmware versions from `dagrom -x`. |
| 9. | Physical layer status reported by:<br>`dagfour` |
| 10. | Network link statistics reported by:<br>`dagfour -si` |
| 11. | Network link configuration from the router where available. |
| 12. | Contents of any scripts in use. |
| 13. | Complete output of session where error occurred including any error messages from DAG tools. The `typescript` Unix utility may be useful for recording this information. |
| 14. | A small section of a captured packet trace illustrating the problem. |

## USE THIS SPACE FOR NOTES

# 5.0 RUNNING DATA CAPTURE SOFTWARE

**Introduction** In running data capture software for a typical measurement session the `scripts/dag42gestart` script is edited and used. The process involves starting a capture session, understanding the high load performance and the DAG card packet transmission capabilities.

**In this chapter** This chapter covers the following sections of information.

- Starting DAG 4.2GE Capture Session
- DAG 4.2GE Card High Load Performance
- DAG Card Packet Transmission Capabilities

## 5.1 Starting DAG 4.2GE Capture Session

**Description** The various tools used for data capture are in the `tools` sub-directory.

For a typical measurement session, first move to the `dag` directory, load the driver, then load the Xilinx receive image to each DAG card.

For example, with one DAG 4.2GE card installed:

```
drv/dagload
tools/dagrom -rvp –d dag0 < xilinx/dag423eth-erf.bit
```

Capture session parameters are set with `dagfour`. The card can operate in two modes, variable length capture (`varlen`), and fixed length capture (`novarlen`).

**Process** Starting a data capture session is described in the following process.

| Process | Description |
|---|---|
| Slen parameter default setting. | Slen parameter is set by default to1536 in `dagfour`.<br><br>This captures the complete content of all standard length packets without wasting bandwidth.<br><br>If only part of a packet is required, such as for IP header capture, the value of `slen` can be changed using `dagfour`.<br><br>`tools/dagfour slen=128 varlen` |

*Continued on next page*

## 5.1 Starting DAG 4.2GE Capture Session, continued

**Process**, continued

| Process | Description |
|---|---|
| Setting variable length mode. | In variable length capture mode, a maximum capture size is set with `slen=N` bytes.<br><br>This figure should be in the range 16 to 1536 and is rounded down to the nearest multiple of 4.<br><br>Packets longer then slen will be truncated.<br><br>Packets shorter than slen will produce shorter records, saving bandwidth and storage space. For full packet capture, for example:<br><br>`tools/dagfour –d dag0 varlen slen=1536` |
| Setting fixed length mode. | In fixed length mode, packets longer than the selected slen will be truncated to slen. Packets shorter than slen will produce records that are padded out to the value of slen.<br><br>large values of slen should not be used in fixed length mode, as short packets arriving will produce large padded records, wasting bandwidth and storage space. E.G. for fixed length 64-byte records, choose slen=44 (64 – ERF header size of 16 – alignment padding 4):<br><br>`tools/dagfour –d dag0 novarlen slen=44` |
| Disabling individual ports. | Each direction [A and B] can be individually enabled and disabled for capture using `dagfour`.<br><br>`tools/dagfour –d dag0 disableb` |

*Continued on next page*

## 5.1 Starting DAG 4.2GE Capture Session, continued

**Process**, continued

| Process | Description |
|---|---|
| Starting a capture session. | Once the capture parameters are configured, a capture session is started by:<br><br>`tools/dagsnap –v –o tracefile`<br><br>Option `-v` provides user information during capture; it can be omitted for automated trace runs.<br><br>If the `–o tracefile` parameter is not specified the tool writes to stdout, which can be used to pipeline `dagsnap` with other tools from `dagtools` package.<br><br>By default dagsnap runs forever. `dagsnap` can be stopped with a signal:<br><br>`killall dagsnap`<br><br>`dagsnap` can also be configured to run for a fixed number of seconds and then exit using the `–s` flag. |

## 5.2 DAG 4.2GE Card High Load Performance

**Description**

As the DAG 4.2GE card captures packets from the network link, it writes a record for each packet into a large buffer in the host PC's main memory.

**Avoiding packet loss**

In order to avoid packet loss, the user application reading the record, such as `dagsnap`, must be able to read records out of the buffer faster than they arrive. Otherwise the buffer eventually fills, and packet records are lost.

For Linux and FreeBSD, when the PC buffer becomes full, the message:

`kernel: dagN: pbm safety net reached 0xNNNNNNNN`"

is displayed on the PC screen, and printed to log `/var/log/messages`.

The "Data capture" LED also goes out. This may be visibly indicated as flashing or flickering.

*Continued on next page*

## 5.2 DAG 4.2GE Card High Load Performance, continued

**Detecting packet losses**

Until some data is read out of the buffer to free some space, any arriving packets subsequently are discarded by the DAG card.

Any loss can be detected in-band by observing the Loss Counter `lctr` field of the Extensible Record Format [ERF]. The Endace ERF is detailed in Chapter 7 of this document.

**Increasing buffer size**

The host PC buffer can be increased to deal with bursts of high traffic load on the network link.

By default the `dagmem` driver reserves 32MB of memory per DAG card in the system. Capture at OC-12/STM-4 (622Mbps) rates and above may require a larger buffer.

128MB or more is suggested for Linux/FreeBSD.

For the DAG 4.2GE card Windows operating system the upper limit is 128MB.

In Debian Linux the amount of memory reserved is changed by editing the file `/etc/modules`.

```
# For DAG 3.x, default 32MB/card
dagmem
#
# For DAG 4.x or 6.x, use more memory per card, E.G.
dagmem dsize=128m
```

The option `dsize` sets the amount of memory used per DAG card in the system.

The value of `dsize` multiplied by the number of DAG cards must be less than the amount of physical memory installed, and less than 890MB.

# 6.0 SYNCHRONIZING CLOCK TIME

**Description**

The Endace DAG range of products come with sophisticated time synchronization capabilities, in order to provide high quality timestamps, optionally synchronized to an external time standard.

The system that provides the DAG synchronization capability is known as the DAG Universal Clock Kit (DUCK).

An independent clock in each DAG card runs from the PC clock. A card's clock is initialised using the PC clock, and then free-runs using a crystal oscillator.

Each card's clock can vary relative to a PC clock, or other DAG cards.

**DUCK configuration**

The DUCK is configured to avoid time variance between sets of DAG cards or between DAG cards and coordinated universal time [UTC].

Accurate time reference can be obtained from an external clock by connecting to the DAG card using the synchronization connector, or the host PCs clock can be used in software as a reference source without additional hardware.

Each DAG card can also output a clock signal for use by other cards.

**Common synchronization**

The DAG card synchronization connector supports a Pulse-Per-Second (PPS) input signal, using RS-422 signalling levels.

Common synchronization sources include GPS or CDMA (Cellular telephone) time receivers.

Endace produces the TDS 2 Time Distribution Server modules and the TDS 6 units that enable multiple DAG cards to be connected to a single GPS or CDMA unit.

More information is on the Endace website, http://www.endace.com/accessories.htm, or the TDS 2/TDS 6 Units Installation Manual.

**In this chapter**

This chapter covers the following sections of information.

- Configuration Tool Usage
- Time Synchronization Configurations
- Synchronization Connector Pin-outs

## 6.1 Configuration Tool Usage

**Description**

The DUCK is very flexible, and can be used in several ways, with or without an external time reference source. It can accept synchronization from several input sources, and can also be made to drive its synchronization output from one of several sources.

Synchronization settings are controlled by the `dagclock` utility.

**Example**

```
dag@endace:~$ dagclock -h
Usage: dagclock [-hvVxk] [-d dag] [-K <timeout>] [-l
<threshold>] [option]

     -h  --help,--usage   this page
     -v  --verbose        increase verbosity
     -V  --version        display version information
     -x  --clearstats     clear clock statistics
     -k  --sync           wait for duck to sync before
                          exiting
     -d  dag              DAG device to use
     -K  timeout          sync timeout in seconds, default
                          60
     -l  threshold        health threshold in ns, default
                          596
  Option:
        default           RS422 in, none out
        none              None in, none out
        rs422in           RS422 input
        hostin            Host input (unused)
        overin            Internal input (synchronize to
                          host clock)
        auxin             Aux input (unused)
        rs422out          Output the rs422 input signal
        loop              Output the selected input
        hostout           Output from host (unused)
        overout           Internal output (master card)
        set               Set DAG clock to PC clock
        reset             Full clock reset.  Load time
                          from PC, set rs422in, none out
```

By default, all DAG cards listen for synchronization signals on their RS-422 port, and do not output any signal to their RS-422 port.

```
dag@endace:~$ dagclock –d dag0
muxin   rs422
muxout  none
status  Synchronized Threshold 596ns Failures 0 Resyncs 0
error   Freq -30ppb Phase -60ns Worst Freq 75ppb Worst Phase 104ns
crystal Actual 100000028Hz Synthesized 67108864Hz
input   Total 3765 Bad 0 Singles Missed 5 Longest Sequence Missed 1
start   Thu Apr 28 13:32:45 2005
host    Thu Apr 28 14:35:35 2005
dag     Thu Apr 28 14:35:35 2005
```

## 6.2 Time Synchronization Configurations

**Description** The DUCK is very flexible, and can be used in several ways, with or without an external time reference source.

The use includes a single card with no reference, two cards with no reference, and a card with reference.

**In this section** This section covers the following topics of information.

- Single Card no Reference Time Synchronization
- Two Cards no Reference Time Synchronization
- Card with Reference Time Synchronization

### *6.2.1 Single Card no Reference Time Synchronization*

**Description** When a single card is used with no external reference, the card can be synchronized to the host PC's clock.

The clock in most PC's is not very accurate by itself, but the DUCK drifts smoothly at the same rate as the PC clock.

If a PC is running NTP to synchronize its own clock, then the DUCK clock is less smooth because the PC clock is adjusted in small jumps. However, overall the DUCK clock does not drift away from UTC.

The synchronization achieved in this case is not as accurate as when using an external reference source such as GPS.

The DUCK clock is synchronized to a PC clock by setting input synchronization selector to overflow:

```
dag@endace:~$ dagclock –d dag0 none overin
muxin   overin
muxout  none
status  Synchronized Threshold 11921ns Failures 0 Resyncs 0
error   Freq 1836ppb Phase 605ns Worst Freq 143377ppb Worst Phase
88424ns
crystal Actual 49999347Hz Synthesized 16777216Hz
input   Total 87039 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Wed Apr 27 14:27:41 2005
host    Thu Apr 28 14:38:20 2005
dag     Thu Apr 28 14:38:20 2005
```

NOTE: `dagclock` should be run only after appropriate Xilinx images have been loaded. If the Xilinx images must be reloaded, the `dagclock` command must be rerun afterwards to restore the configuration.

### 6.2.2 Two Cards no Reference Time Synchronization

**Description**

When two DAG cards are used in a single host PC with no reference clock, the cards are to be synchronized in some way if timestamps between the two cards are to be compared. For example, if two cards monitor different directions of a single full-duplex link.

Synchronization between two DAG cards is achieved in two ways. One card can be a clock master for the second, or one can synchronize to the host and also act as a master for the second.

**Synchronizing cards**

If both cards are to be accurately synchronized, but not so for absolute time of packet time-stamps being correct, then one card is configured as the clock master for the other.

**Locking cards together**

Although the master card's clock will drift against UTC, the cards are locked together.

The cards are locked together by connecting the synchronization connector ports of both cards with a standard RJ-45 Ethernet cross-over cable.

Configure one of the cards as the master, the other defaults to being a slave.

```
dag@endace:~$ dagclock –d dag0 none overout
muxin   none
muxout  over
status  Not Synchronized Threshold 596ns Failures 0 Resyncs 0
error   Freq 0ppb Phase 0ns Worst Freq 0ppb Worst Phase 0ns
crystal Actual 100000000Hz Synthesized 67108864Hz
input   Total 0 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Thu Apr 28 14:48:34 2005
host    Thu Apr 28 14:48:34 2005
dag     No active input - Free running
```

The slave card configuration is not shown, the default configuration is sufficient.

*Continued on next page*

## 6.2.2 Two Cards no Reference Time Synchronization, continued

**Preventing time-stamps drift**

To prevent the DAG card clocks time-stamps drifting against UTC, the master can be synchronized to the host PC's clock which in turn utilises NTP. This then provides a master signal to the slave card.

The cards are locked together by connecting the synchronization connector ports of both cards with a standard RJ-45 Ethernet cross-over cable.

Configure one card to synchronize to the PC clock and output a RS-422 synchronization signal to the second card.

```
dag@endace:~$ dagclock –d dag0 none overin overout
muxin   over
muxout  over
status  Synchronized Threshold 11921ns Failures 0 Resyncs 0
error   Freq -691ppb Phase -394ns Worst Freq 143377ppb Worst Phase
88424ns
crystal Actual 49999354Hz Synthesized 16777216Hz
input   Total 87464 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Wed Apr 27 14:27:41 2005
host    Thu Apr 28 14:59:14 2005
dag     Thu Apr 28 14:59:14 2005
```

The slave card configuration is not shown, the default configuration is sufficient.

## 6.2.3 Card with Reference Time Synchronization

**Description**

The best timestamp accuracy occurs when DAG card is connected to an external clock reference, such as a GPS or CDMA time receiver.

**Pulse signal from external sources**

The DAG synchronization connector accepts a RS-422 Pulse Per Second [PPS] signal from external sources.

This is derived directly from a reference source, or distributed through the Endace TDS 2 [Time Distribution Server] module which allows two DAG cards to use a single receiver.

More cards can be accommodated by daisy-chaining TDS-6 expansion units to the TDS-2 unit, each providing outputs for an additional 6 DAG cards.

*Continued on next page*

## *6.2.3 Card with Reference Time Synchronization*, continued

**Using external reference source**

To use an external clock reference source, the host PC's clock must be accurate to UTC to within one second. This is used to initialise the DUCK.

The external time reference allows high accuracy time synchronization.

When the time reference source is connected to the DAG synchronization connector, the card automatically synchronizes to a valid signal.

```
dag@endace:~$ dagclock –d dag0
muxin rs422
muxout none
status Synchronized Threshold 596ns Failures 0 Resyncs 0
error Freq 30ppb Phase -15ns Worst Freq 2092838ppb Worst Phase 33473626ns
crystal Actual 100000023Hz Synthesized 67108864Hz
input Total 225 Bad 0 Singles Missed 1 Longest Sequence Missed 1
start Thu Apr 28 14:55:20 2005
host Thu Apr 28 14:59:06 2005
dag Thu Apr 28 14:59:06 2005
```

**Connecting time distribution server**

The TDS 2 module connects to any DAG card with a standard RJ-45 Ethernet cable and can be placed some distance from a DAG card.

Existing RJ-45 building cabling infrastructure can be used to cable synchronization ports.

CAUTION: Never connect a DAG card and/or the TDS 2 module to active Ethernet or telephone equipment.

**Testing signal**

For Linux and FreeBSD, when a synchronization source is connected the driver outputs some messages to the console log file `/var/log/messages`.

The `dagpps` tool is used to test a signal is being received correctly and is of correct polarity. To perform the test, run:

```
dagpps –d dag0.
```

The tool measures input state many times over several seconds, displaying polarity and length of input pulse.

Some DAG cards have LED indicators for synchronization (PPS) signals.

## 6.3 Synchronization Connector Pin-outs

**Description** DAG cards have an 8-pin RJ45 connector with two bi-directional RS422 differential circuits, A and B. The PPS signal is carried on circuit A, and the serial packet is connected to the B circuit.

**Pin assignments** The 8-pin RJ45 connector pin assignments are:

| | |
|---|---|
| 1. | Out A+ |
| 2. | Out A- |
| 3. | In A+ |
| 4. | In B+ |
| 5. | In B- |
| 6. | In A- |
| 7. | Out B+ |
| 8. | Out B- |

**Figure** Figure 6-1 shows the RJ45 plug and socket connector pin-outs.

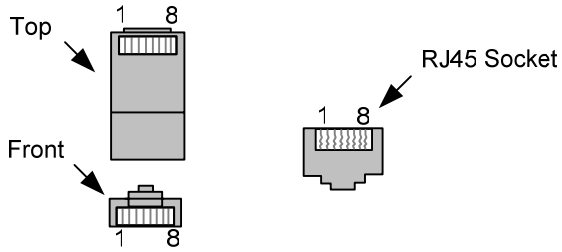



Figure 6-1. RJ45 Plug and Socket Connector Pin-outs.

**Out-pin connections** Normally the GPS input should be connected to the A channel input, pins 3 and 6. The DAG can also output a synchronization pulse; used when synchronizing two DAG's without a GPS input. Synchronization output is generated on the Out A channel, pins 1 and 2.

**Ethernet crossover cable** A standard Ethernet crossover cable can be used to connect the two cards.

| | | | |
|---|---|---|---|
| TX_A+ | 1 | 3 | RX_A+ |
| TX_A- | 2 | 6 | RX-A- |
| RX_A+ | 3 | 1 | TX_A+ |
| RX_B+ | 4 | 7 | TX_B+ |
| RX_B- | 5 | 8 | TX_B- |
| RX_A- | 6 | 2 | TX_A- |
| TX_B+ | 7 | 4 | RX_B+ |
| TX_B- | 8 | 5 | RX_B- |

**Support** For cables and further advice on using GPS and CDMA time receivers email support@endace.com.

## USE THIS SPACE FOR NOTES

# 7.0 DATA FORMATS OVERVIEW

**In this chapter** This chapter covers the following sections of information.

- Data Formats
- Timestamps

## 7.1 Data Formats

**Description** The DAG card uses the ERF Type 2 Ethernet Variable Length Record. Timestamps are in little-endian [Pentium native] byte order. All other fields are in big-endian [network] byte order. All payload data is captured as a byte stream, no byte re-ordering is applied.

**Table** Table 7-1 shows the generic variable length record. The diagram is not to scale.

<table>
<tr><td colspan="3">timestamp</td></tr>
<tr><td colspan="3">timestamp</td></tr>
<tr><td>type</td><td>flags</td><td>rlen</td></tr>
<tr><td colspan="2">lctr</td><td>wlen</td></tr>
<tr><td colspan="3">(rlen - 16) bytes of record</td></tr>
</table>

Table 7-1. Generic Variable Length Record.

**Data format** The following is an overview of the data format used.

| Data Format | Description |
|---|---|
| type: | This field contains an enumeration of the frame subtype. If the type is zero, then this is a legacy format.<br><br>0: TYPE_LEGACY<br>1: TYPE_HDLC_POS: PoS w/HDLC framing<br>2: TYPE_ETH: Ethernet<br>3: TYPE_ATM: ATM Cell<br>4: TYPE_AAL5: reassembled AAL5 frame<br>5: TYPE_MC_HDLC: Multi-channel HDLC frame<br>6: TYPE_MC_RAW: Multi-channel Raw link data<br>7: TYPE_MC_ATM: Multi-channel ATM Cell |

*Continued on next page*

## 7.1 Data Formats, continued

**Data format**, continued

| Data Format | Description |
|---|---|
| flags: | This byte is divided into 2 parts, the interface identifier, and the capture offset.<br><br>1-0: capture interface 0-3<br>2: varying record lengths present<br>3: truncated record [insufficient buffer space]<br>4: rx error [link error]<br>5: 5: ds error [internal error]<br>7-6: reserved |
| Rlen: record length | Total length of the record transferred over PCI bus to storage. |
| Lctr: *loss counter* | A 16 bit counter, recording the number of packets lost since the previous record. Records can be lost between the DAG card and memory hole due to overloading on PCI bus. The counter starts at zero, and sticks at 0xffff. |
| Wlen: *wire length* | Packet length including some protocol overhead. The exact interpretation of this quantity depends on physical medium. |
| offset: | Number of bytes *not* captured from start of frame.<br><br>Typically used to skip link layer headers when not required in order to save bandwidth and space.<br><br>This field is currently not implemented, contents can be disregarded. |

*Continued on next page*

## 7.1 Data Formats, continued

**Table**

Table 7-2 shows the Type 2 Ethernet variable length record. The diagram is not to scale.

| | | |
|---|---|---|
| timestamp | | |
| timestamp | | |
| type:2 | flags | rlen |
| lctr | | wlen |
| offset | pad | rlen-18 |
| bytes of frame | | |

Table 7-2. Type 2 Ethernet Variable Length Record.

The Ethernet frame begins immediately after the pad byte so that the layer 3 [IP] header is 32Bit-aligned.

## 7.2 Timestamps

**Description**

The ERF format incorporates a hardware generated timestamp of the packet's arrival.

The format of this timestamp is a single little-endian 64-bit fixed point number, representing seconds since midnight on the first of January 1970.

The high 32-bits contain the integer number of seconds, while the lower 32-bits contain the binary fraction of the second. This allows an ultimate resolution of $2^{-32}$ seconds, or approximately 233 picoseconds.

Another advantage of the ERF timestamp format is that a difference between two timestamps can be found with a single 64-bit subtraction. It is not necessary to check for overflows between the two halves of the structure as is needed when comparing Unix time structures, which are also available to Windows users in the Winsock library.

Different DAG cards have different actual resolutions. This is accommodated by the lowermost bits that are not active being set to zero. In this way the interpretation of the timestamp does not need to change when higher resolution clock hardware is available.

*Continued on next page*

## 7.2 Timestamps, continued

**Description**,continued

**Example code**

Here is some example code showing how a 64-bit ERF timestamp (erfts) can be converted into a struct timeval representation (tv).

```
unsigned long long lts;
struct timeval tv;

 lts = erfts;
 tv.tv_sec = lts >> 32;
 lts = ((lts & 0xffffffffULL) * 1000 * 1000);
 lts += (lts & 0x80000000ULL) << 1;      /* rounding */
 tv.tv_usec = lts >> 32;
 if(tv.tv_usec >= 1000000) {
    tv.tv_usec -= 1000000;
    tv.tv_sec += 1;
      }
```